## Module Five

## Security Policies and Security Policy Models

This module introduces the concepts of a security policy and a securitypolicy model. It examines the issues that they address, illustrates the relationships between them, and shows how they are used to help develop trustedsystems . Several security policy models are mentioned, including the Bell-LaPadula, Biba, and Clark-Wilson models.

## Module Learning Objectives

The material presented in this module can be read independently of theother modules. Upon completion of this module, the student should:

1. Understand the value and importance of an explicitly stated security policy.
2. Understand the role played by the system security policy model in trusted systems development.
3. Understand the components of state-machine models and how they are used to describe system security.
4. Be familiar with techniques to express state-machine models and techniques to show implementation correspondence to the model.
5. Be familiar with various types of security policy models.

## Overview

Security policies and security policy models are usually discussedtogether because it is often difficult to determine where a security policy endsand where a model begins. This module begins by describing how requirements for information protection must be accurately captured in a system securitypolicy. It then describes how a security policy model captures the implementationof aspects of a security policy on a given system. Security policy modelingis described based on the state-machine approach.

## Security Policies

A *security policy* establishes accountability for information protection by defining a set of rules, conditions, and practices that regulate how an organization manages, protects, and distributes sensitive information. While substantial effort may be expended by the vendor in implementing the *mechanisms* to enforce the policy and developing *assurance* that the mechanisms perform properly, all is for naught if the policy itself is flawed or poorly understood. For this reason, the TCSEC requires that "there must be an explicit and well-defined security policy enforced by the system."

A security policy may address confidentiality, integrity, and/or availability. The TCSEC requires a policy that focuses primarily on confidentiality. Clark and Wilson [Clark87] describe an integrity policy not explicitly required bythe TCSEC, but vendors may find that this policy captures requirements frequently desired by end users, including government users. Little, if any, work has been done on availability policies.

The standard features that are required to support a TCSEC-like security policy are clearances for users, classifications for information, and controls of user access to information based on comparisons between the user's clearance and the information's classification. For example, users are granted clearances (e.g., Secret, Top Secret) and information is classified according to the sensitivity of the information (again Secret, Top Secret, etc.). Clearances and classifications can also include additional constraints (categories, compartments, markings or handling caveats), such as "NATO", "No Foreign", or codewords for specific projects. Users typically must be a member of a specific category or compartment in addition to being cleared for the sensitivity of the information before access can be granted. In addition to having the requisite clearance, the TCSEC policy requires users to have need-to-know for information. In other words, just because a user has an appropriate clearance does not mean the user has the right to access particular information.The user must also have a need to access the information based on the need to do his job. More detail on these policy areas is presented in Modules 8 and 9, which cover mandatory access control (MAC) and discretionary access control(DAC), respectively. Other areas of a TCSEC-like security policy include: object reuse (see Module 10), identification and authentication (see Module11), and audit (see Module 12).

Typically, the security policy for an automated information system (AIS) destined for the U.S. Government or one of its contractors will have its roots in laws, regulations, and/or directives that apply to the customer (e.g., Executive Order 12356 [NSI82], DoD Directive 5200.1-R [ISPR82], DoD Directive 5200.28 [AIS88], DCID 1/16 [SFI88]). These documents may be supplemented and/or interpreted by additional Military Department directives (e.g., Air Force Regulation 205-16 [SPPR84], Department of the Navy OPNAV Instruction 5239.1 [NAVSP85], DIA Manual 50-4 [SCCO80]). The requirements of these policy documents, together with any local security requirements, should be interpreted for a specific AIS and incorporated into an explicitlystated AIS security policy. The clear definition of applicable requirements is necessary for the development of a security policy model.

## Security Policy Models

A *security policy model* precisely and unambiguously conveys those aspects of the security policy that are enforced by the system. It is an abstractionof the security policy that is stated in terms of operational entities (e.g., subjects and objects). A security policy model is typically interpreted in moreconcrete terms to apply to a specific system. For example, the Bell-LaPadula model is an abstract representation of the U.S. Government security policy; the Multics interpretation of the Bell-LaPadula model is a specific, concrete representation of the Bell-LaPadula model for the Multics operating system. While class B2 and higher requires a formal, mathematically stated model, models for class B1 may be informally written in structured English. In either case, the model is intended to serve as a paradigm of system security. It clearly states what it means for the AIS to be secure. It should be simple, unambiguous, and sufficiently abstract to be an obvious representation of the securitypolicy enforced by the AIS. Indeed, the model's simplicity and intuitive appeal are the principle means of ensuring its correct presentation of the requirementslevied by governing regulations.

The State-Machine Approach

While other promising techniques to modeling exist (e.g., temporal logic, denotational semantics, algebraic specification), the state-machine concept is so pervasive that everyone doing modeling work should understand it [Gasser88]. State-machine models represent AIS operations as changes in value of a set of *state variables* in response to *rules of operation*. In order to describe the model, it is necessary to identify all the state variablesand their initial values (the *initial state*), specify the results of the rules of operation, and state required security properties. Only those values which are pertinent to the security of the system are selected as state variables. Similarly only those AIS commands which can affect AIS security are described in the rules of operation.

For example, in modeling an AIS, login might be described as a rule of operation since it can provide a user access to the AIS and assign theuser rights and privileges after successfully establishing the user's identity and authorization. Sample state variables used by this rule of operationmight include user id, user clearance, session working label, etc.

*Security properties* describe what relationships must exist between the values of state variables in order to enforce the security requirementscontained in the policy statement. For instance, it may be required that the session working label always be less than or equal to the user's clearance (i.e., the user may not initiate a session at a sensitivity label for which he or she does not possess a personal clearance), which implies something about the login rule of operation. There are three types of properties commonly used to express security requirements:

1. invariants
2. constraints
3. information flow requirements

Invariants are requirements which refer to individual states of the system. The term invariant comes from the fact that these are conditions which must be true in every state the system can reach (their truth does not vary). An example invariant would be a requirement that users may only possess access to information for which they are cleared. Invariants are used to define a secure state of the system, and the proof that the invariants defining asecure state are true in every reachable state proves that the system will always be in a secure state.

There are some security properties which cannot be described in terms of conditions on individual states. Many of these can be described as conditions on pairs of consecutive states. Such properties are called constraints. An example of a constraint would be a requirement that sensitivity labelsof objects cannot be changed except by the Information System Security Officer (ISSO). A restatement of this requirement is: if the invoker of anoperation is not the ISSO, then the sensitivity label of the object after the operation equals the sensitivity label of the object before the operation. In this form,the property is clearly a constraint on the relation of values before and after the operation.

Other properties are stated in terms of restrictions on allowedinformation flows among variables defining the state machine. An operation is said to cause information flow from variable A to variable B if the value of B afterthe operation depends in some way on the value of A before the operation. The typical information flow requirement is: information can only flowfrom A to B if the sensitivity label of B dominates the sensitivity label of A.Information flow properties can take the form of constraints in the above sense, or they can be phrased as properties about I/O histories or state histories. Constraints can be demonstrated for each operation individually. When information-flow properties are given as properties of histories, “unwinding” theorems are often used to recast them as constraints.

To prove that an invariant is true in every state of a state-machine model, the modeler must:

1. specify the initial state of the system (most likely conditions immediately following system boot) and show that the invariant is satisfied in the initial state, and
2. given that the AIS is in a secure state, show that the state which results from the application of each rule of operation satisfies the invariant.

It is the modeler’s responsibility to ensure that all the necessary security properties are described to represent the notion of security conveyed bythe governing security policy documents. It is also the modeler’s responsibility to completely describe the security-relevant operation of the AIS in the appropriate number of rules of operation. The modeler must alsodemonstrate the internal consistency of the model as described above. For formal models, this step is performed most rigorously by the application of formal tools.

Other approaches to AIS security modeling have been formulated in order to achieve greater precision and/or generality. Information-flow is one approach in particular that is motivated by an informal view of how informationflows through a deterministic state-machine system. An introduction to information-flow models, including the well-known Goguen-Meseguer non-interference model [Goguen82], is given in [MODEL92, Sec. 3.2] and [Williams91].

<u>Confidentiality Models</u>

Perhaps the best-known state-machine model of government security requirements was formulated by David Bell and Leonard LaPadula and applied to the Multics trusted computer system [Bell76]. The TCSECdefines the Bell-LaPadula model as:

> “A formal state transition model of computer security policy that describes a set of access control rules. In this formal model, theentities in a computer system are divided into abstract sets of subjects and objects. The notion of a secure state is defined and it is proven thateach state transition preserves security by moving from secure state tosecure state; thus, inductively proving that the system is secure. A systemstate is defined to be “secure” if the only permitted access modes ofsubjects to

objects are in accordance with a specific security policy. In order to determine whether or not a specific access mode is allowed, the clearance of a subject is compared to the classification of the objectand a determination is made as to whether the subject is authorized for the specific access mode. The clearance / classification scheme is expressed in terms of a lattice."

The Bell-LaPadula model has been interpreted and adapted to serve as the security policy model for many TCSEC-based systems. Frequently, reference to the Bell-LaPadula model is really directed to the kinds of security properties it formulates as opposed to the model as a whole, which includes rules of operation and proofs as well as security properties. Frequently referenced security properties which capture the basics of government classified information disclosure protection include:

- *Simple Security Property* -- A subject may not have current observe access to an object unless the subject's clearance dominates the object's classification.
- **-Property* (pronounced Star Property) -- A subject may not have current alter access to an object unless the object's classification dominates the subject's clearance.
- *Discretionary Security Property* -- A subject may not have current access to an object in a given access mode unless discretionary permission has been explicitly granted for the subject to access that object in the specified mode. (In this form, withdrawal of discretionary permission does not necessitate immediate revocation of access; [Bell76] models immediate revocation, and [Karger89] shows how to implement it, but immediate revocation is not required.)
- A less frequently referenced and quite restrictive property is the *Tranquillity Property*, which states that a subject's clearance and an object's classification cannot be changed. Some variant of tranquillity, however, is necessary in order to ensure security.

The TCSEC MAC policy translates into two basic rules when enforced by a computer system, "no read-up" and "no write-down." No read-up prevents users from accessing information for which they are not cleared to access. No write-down prevents users (or more importantly software) from taking more sensitive information and writing it into a less sensitive document(or other information store). The above simple-security and *-properties are acomputer -oriented recasting of these two rules.

Any security model must be shown (informally) to describe anappropriate policy for the modeled system. Since models based on the Bell-LaPadula model have been widely used for basic government mandatory and discretionary policies, showing a model includes the above kinds of security properties is one way to show the model describes an acceptable government classified information disclosure policy.

Integrity Models

A taxonomy of integrity policies and models is given in [Roskos90]. Some integrity models are similar to the Bell-LaPadula model (e.g., Biba integrity model [Biba77]), as are some that combine integrity andconfidentiality (Dion' s protection model [Dion81], secure distributed data view model[Lunt89]). Others, such as the Clark-Wilson model [Clark87], differ significantly.

To enforce the Clark-Wilson commercial security policy, the system must control not only user access to programs, but also program access to data. For example, to enforce two-person controls for a specific transaction, two programs can be written, one for each half of an operation, and the use of these two programs must be restricted to specific (distinctlydifferent) individuals . To enforce this two-person control, the system must be able to restrictspecific programs to specific operations on specific pieces of data. Systemsbuilt to TCSEC requirements typically treat programs as data, and as such, access to these programs can be controlled in a manner equivalent to what isrequired in a commercial environment. However, once a program is invoked, TCSEC systems typically restrict the program's access to data based on the identity of the user executing the program rather than the identity of the programitself .

Availability Models

Models of availability have concentrated primarily on denial of service. In [Millen92], for example, a resource-allocation model is given that facilitates discussion of general availability requirements such as the following:

*Finite Waiting Time* - in any given state, each subject (e.g., process) eventually runs in some future state.

*Maximum Waiting Time* - there is a maximum "waiting" time, W, such that in any given state, there is, within time W, a future state inwhich a process is running.

## Relevant Trusted Product Evaluation Questionnaire Questions

### 2.1 SUBJECTS

A subject is an active entity in the system, generally in the form of aprocess or device that causes information to flow among objects or changes the system state. In Multics, a subject was viewed as a process/domain pair whose access controls were checked prior to granting access to objects.

C1:

1. (a) List and (b) describe the subjects in your system?
2. (a) When and (b) how are the subjects created? (Forexample, they can be created or activated when a user logs on or when a process is spawned.)
3. (a) When and (b) how are the subjects destroyed? (Forexample, they can be destroyed or deactivated when a process terminates or when the user logs off.)

4. (a) What are the security attributes of a subject? (Examples of security attributes are user name, group id, sensitivity level, etc.) For each type of subject in your system (i.e., process, device,etc.), what mechanisms are available to (b) define and (c) modifythese attributes? (d) Who can invoke these mechanisms?

5. (a) What are other privileges a subject can have? (Examples of such privileges are: super user, system operator, system administrator, etc. Your operating system may assign numerous other privileges to the subjects, such as the ability to use certain devices.) For each type of subject in your system, what mechanisms are available to (b) define and (c) modify these privileges? (d) Who can invoke these mechanisms? (e) Providea list of subjects within the TCB boundary and (f) the list of privileges for each of them.

6. When a subject is created, where do its (a) security attributesand (b) privileges originate, i.e., how are the security attributesand privileges inherited?

7. List the subjects, if any, which are not controlled by the TCB.

## 2.2 OBJECTS

An object is a passive entity that contains or receives information.Access to an object potentially implies access to the information it contains. Examples of objects are: records, blocks, pages, segments, files, directories, directory trees, and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes.

C1:

1. Provide a list of objects controlled by

## 2.12 MODELING AND ANALYSIS

B1:

1. Describe the system security policy.

2. How is the system security policy represented in the security policy model?

3. What policies are represented in the model (e.g., MAC, DAC, privileges, other protection mechanisms, object reuse)?

4. What tools, techniques, and methodologies are used to demonstrate that the model is consistent with its axioms? That is, what evidence is offered to show that the model's definition of security will be enforced, assuming that the rules of operation are faithfully implemented?

B2:

6. What tools, techniques and methodologies are used to represent the formal model of the system security policy?

7. What policies are represented in the formal model (e.g., MAC, DAC, privileges, other protection mechanisms, object reuse)?

8. What tools, techniques, and methodologies are used to prove the model consistent with its axioms? That is, what proof is offered that the model's definition of security will be enforced, assuming that the rules of operation are faithfully implemented?

## Required Readings

TCSEC85 National Computer Security Center, *Department of Defense Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD, December 1985.

Sections 2.1.1, 2.2.1, 3.1.1, 3.2.1, 3.3.1, and 4.1.1, contain the security policy requirements. Sections 3.1.3.2.2, 3.2.3.2.2, 3.3.3.2.2, and 4.1.3.2.2 contain the security policy model requirements, which are summarized on pages 98-99.

INTERP94 National Computer Security Center, *The Interpreted TCSEC Requirements*, (quarterly).

The following Interpretations are relevant to security policies:

| | |
|---|---|
| I-0002 | Delayed revocation of DAC access |
| I-0020 | DAC authority for assignment |
| I-0022 | One set of banner pages around multiple outputs |
| I-0039 | Multilevel printers and page labeling |
| I-0040 | Requirements for overwrite label capability |
| I-0041 | Object reuse applies to all system resources |
| I-0239 | Subject access revocation after change in user clearance |
| I-0275 | Single-level printers and page labeling |
| I-0312 | Set-ID and the DAC requirement |
| C1-CI-05-84 | Exportation to Multilevel Devices |
| C1-CI-06-84 | Discretionary Access Control |
| C1-CI-01-85 | Device Labels |
| C1-CI-03-85 | Discretionary Access Control |
| C1-CI-01-86 | Discretionary Access Control |
| C1-CI-03-86 | DAC by Default |
| C1-CI-01-88 | Exportation of Labels |
| C1-CI-03-89 | DAC Public Objects |

None of the Interpretations are relevant to security policy models.

Gasser88 Gasser, M., *Building a Secure Computer System*, Van Nostrand Reinhold Co., N.Y., 1988.

Chapter 9 addresses security models and should be studied in its entirety. It provides excellent introductory coverage of nearly all of the topics included in the topic outline and is very readable. Section 6.5 presents the Biba integrity model.

MODEL92 National Computer Security Center, *A Guide to Understanding Security Modeling in Trusted Systems*, NCSC-TG-010, Version 1, October 1992.

This document was written to provide guidance to developers, evaluators, and users of security policy models. It overviews the security modeling process, discusses the modeling of various kinds of policies and objectives, presents security modeling techniques, and reviews the TCSEC security model requirements. The first 65 pages should be read in their entirety for this module.

## Supplemental Readings

None.

## Other Readings

The literature is rich with example models for all sorts of applications that the modeler may draw upon; many of the better-known models are discussed and referenced in [MODEL92].

AIS88 Department of Defense, *Security Requirements for Automated Information Systems*, DoD 5200.28, March 1988.

Bell76 Bell, D.E. and La Padula, L.J., *Secure Computer Systems: Unified Exposition and Multics Interpretation*, MTR-2997, Rev. 1, MITRE Corporation, Bedford, MA, March 1976.

Biba77 Biba, K.J., *Integrity Considerations for Secure Computer Systems*, MTR-3153, MITRE Corporation, Bedford, MA, 1977.

Clark87 Clark, D.D. and Wilson, D.R., “A Comparison of Commercial and Military Computer Security Policies,” *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pp. 184-194, May 1987.

Dion81 Dion, L.C., “A Complete Protection Model,” *Proceedings of the 1981 IEEE Symposium on Security and Privacy*, pp. 49-55, April 1981.

Goguen82 Goguen, J.A. and Meseguer, J., “Security Policies and Security Models,” *Proceedings of the 1982 Symposium on Security and Privacy*, pp. 75-86, April 1982.

ISPR82 Department of Defense, *Information Security Program Regulation*, DoD 5200.1-R, August 1982.

Karger89 Karger, P.A., “New Methods for Immediate Revocation,” *Proceedings of the 1989 IEEE Symposium on Security and Privacy*, pp. 48-55, May 1989.

Lunt86 Lunt, T., Neumann, P., Denning, D., Schell, R., Heckman, M., and Shockley, W., *Secure Distributed Data Views, Vol. 1: Security Policy and Interpretation for a Class A1 Multilevel Secure*

*Relational Database System*, prepared for USAF RADC under contract F30602-85-C-0243, November 1986.

Lunt89 Lunt, T. Denning, D., Schell, R., Heckman, M., and Shockley, W., *The Seaview Formal Security Policy Model*, prepared for USAF RADC under contract F30602-85-C-0243, February 1989.

Millen92 Millen, J. K., "A Resource Allocation Model for Denial ofService," *Proceedings of the 1992 IEEE Symposium on Research in Security and Privacy*, pp. 137-147, May 1992.

NAVSP85 Office of the Chief of Naval Operations, *Department of the Navy Automatic Data Processing Security Program*, OPNAV Instruction 5239.1A, Washington, DC, April 1985.

NSI82 The Office of the President, "Executive Order 12356 of April 2, 1982: National Security Information," *Federal Register*, Vol. 47 No. 66, Washington, DC, 1982.

Roskos90 Roskos, J.E., Welke, S.R., Boone, J., and Mayfield, T., "A Taxonomy of Integrity Models, Implementations and Mechanisms," *Proceedings of the 13th National Computer Security Conference*, pp. 541-551, October 1990.

SCCO80 Defense Intelligence Agency, *Security of Compartmented Computer Operations (U) (CONFIDENTIAL)*, DIAM 50-4, Washington, DC, June 1980.

SFI88 Director of Central Intelligence, *Security of Foreign Intelligence in Automated Data Processing Systems and Networks (U) (SECRET)*, DCID 1/16, Washington, DC, revised 1988.

SPPR84 Headquarters U.S. Air Force, *Automatic Data Process (ADP) Security Policy, Procedures, and Responsibilities*, AF Regulation 205-16, Washington, DC, August 1984.

Williams91 Williams, J. G., "Modeling Nondisclosure in Terms of the Subject-Instruction Stream," *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*, pp. 64-77, May 1991.